

SYSTEM AND MEHTOD OF INTERFACING FOR CLIENT
APPLICATION PROGRAMS TO ACCESS
A DATA MANAGEMENT SYSTEM

5 **PRIORITY**

This application claims priority to applicants' co-pending application entitled "Computer System for Data Management and Method for Operating said System" having Serial No. 08/836,713, filed May 13, 1997, claiming priority to applicants' PCT application No. 10 PCT/EP96/03972, filed September 10, 1996, which claims priority to applicants' EPO application No. 95114467.4 filed September 14, 1995 (hereinafter the "CATS-OS system"), each of which is incorporated herein by this reference. This application also claims priority to co-pending application entitled "Computer System for Data Management and Method for Operating said System" having Serial No. 09/454,978, filed December 15 3, 1999, which claims priority to co-pending application entitled "Computer System for Data Management and Method for Operating said System" having Serial No. 08/836,713, filed May 13, 1997, and applicants' applications entitled "Use of Computer System for Data Management and Method for Operating the System for Request for Quote" having Serial No. 20 60/111,031, "Sales Trader Use of Computer System for Data Management and Method for Operating the System" having Serial No. 60/111,030, and "Method and System for Warrant Trading Data Management" having

09602588 601500

Serial No. 60/111,032, each filed December 4, 1998, each of which is also incorporated herein by this reference.

FIELD OF THE INVENTION

5 The present invention is a system and method for allowing trading of securities using a network and at least one computer that dynamically creates multiple classes to execute trading functions.

BACKGROUND OF THE INVENTION

10 With the advent of technology has come the desire to execute securities transactions in faster and more efficient methods. One method involves using a global network and a computer system as an interface between the person desirous of executing the trade and the trader.

15 The existing CATS-OS system includes use of a client-server architecture, which has a number of servers. These servers include, for example, a transaction server, a rate server, a hand-off server, a credit server, and a security server. In a typical interaction with the system, the user of the CATS-OS system connects to the system and is authenticated by the security manager, which allows the user certain privileges depending 20 on the level of the user, such as customer, trader, and administrator.

 A request for quote aspect of the CATS-OS system allows deals to be consummated that exceed the limits allowed by the host bank and in other circumstances in which a rate may not be available, for example, by allowing a trader to respond to a request for a quote and to provide a

0 9 6 6 2 5 6 8 8 0 9 4 2 0 0

proposed price against which the user can trade. A sales trader aspect of the CATS-OS system allows a special type of trader, referred to as a sales trader, to deal on behalf of a selected customer within the CATS-OS system. A multi-bank aspect of the CATS-OS system allows a plurality of banks to deliver their prices and enables the single bank CATS-OS system to perform in a multi-bank mode by establishing links to one or more additional banks.

In the existing CATS-OS system, every component of the CATS-OS object linking and embedding (OLE) interfaces has only one top-level single createable class, called master class. The existing version of the OLE servers in the CATS-OS OLE interface has a simple flat hierarchy. Presently, the instance properties of all of its classes are set to be createable multi-use so that the OLE server can handle multiple objects of every class created by one or more OLE client applications. With this structure, only one instance of CATS-OS OLE interface is running to deal with all connections. Obviously, the existing CATS-OS interface can become a bottleneck when massive parallel connection requests are invoked.

A goal of the existing CATS-OS system is to provide the ability for a user working at a workstation, or particularly at a personal computer (PC), to automatically and electronically trade whether it is foreign exchange (FX) warrants or other instruments. As the existing CATS-OS system has been deployed to customers, and as electronic commerce has grown, it has become apparent that additional issues arise in doing electronic commerce.

For example, an issue that many organizations have is that in performing a trade at a PC, redundant entries are necessary into a number of the organization's other systems. Another potential issue for certain organizations which wish to use the existing CATS-OS system for automating electronic commerce is that a certain degree of manual intervention is still required for a user to perform the trade at the PC, so it is difficult to thoroughly automate the process.

In one exemplary conventional system, the server computer has installed into its software a single class. A class is a specification of what types of data and functions are permissible. As an example, one class may involve letter manipulation. In this class, the data type is "string" which is a serial collection of letters. A permissible function within this class is addition. Thus, a user of this class first creates objects (i.e. words like "cat" and "s") and adds them together (i.e. "cat" + "s" results in the new string "cats").

In this example, the addition function is defined to add the second presented string to the first in a serial fashion. Thus, if a user were to execute this addition function as follows: "s" + "cat", the result would be a new string "scat". Similarly, if the data type were strictly limited to letters, the function "4" + "7" would not result in "47" nor the traditional arithmetic result "11" but would result in an error message as the class would be expecting to operate on letters and not numbers.

From this simple example, a class is therefore defined as a software tool that allows users to operate on 1) specific data types using 2) specifically defined functions.

With this background established, a conventional trading system
5 was developed where a single class was defined and the multiple traders accessed the definitions and functions of that class. While a single class can operate on multiple objects of that class, a bottleneck occurs if too many objects are in need of class resources at one time. More specifically, in this conventional system, each trader in the exchange is given his/her
10 own object of the class. If there are thousands of traders trying to execute trades simultaneously, the computer will continually access the class definitions and functions to execute those trades. If there are more objects accessing class resources than the system can process at one time, the computer will begin to generate error messages and prevent certain trades
15 from being executed. In a very bad scenario, the computer may shut down from this overload.

SUMMARY OF THE INVENTION

The present invention overcomes the problems of the conventional systems by creating multiple classes to execute a plurality of trades in parallel. The number of classes created dynamically at any given time is proportional to the number of clients attempting to execute securities transactions at that time. In addition, in another preferred embodiment of the present invention, each class created, has the capability of processing a
20

plurality of objects at a single time. Thus, the capabilities of the software are expanded and the number of clients who can execute securities transactions at any given time is increased.

5 BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and form a part of the specification, illustrate preferred embodiments of the present invention and, together with the description, disclose the principles of the invention. In the drawings:

10 Figure 1 is a block diagram of a system, according to a preferred embodiment of the present invention;

Figure 2 is a screen-shot of a Web site, according to a preferred embodiment of the present invention;

15 Figure 3 is a screen-shot of a Web site, according to a preferred embodiment of the present invention;

Figure 4 is a screen-shot of a Web site, according to a preferred embodiment of the present invention;

Figure 5 is a flowchart of a process for executing a trade, according to a preferred embodiment of the present invention; and

20 Figure 6 is a block diagram of components within a computer, in accordance with a preferred embodiment of the present invention.

DETAILED DESCRIPTION OF THE DRAWINGS

Reference will now be made in detail to preferred embodiments of the invention, non-limiting examples of which are illustrated in the accompanying drawings. Figure 1 is a system 100 in accordance with an embodiment of the present invention. A customer 105 uses his/her personal computer (PC) 110 to interact with the other components of system 100. It should be noted that in alternative embodiments, PC 110 is replaced with either a cell phone, a personal digital assistant or pager that all have Web browsing capabilities.

5 PC 110 is coupled to a network 115. In a preferred embodiment, network 115 is the Internet. Network 115 is coupled to a computer 120. Computer 120 is coupled to database 125. Computer 120 is also coupled to another network 130, which, in a preferred embodiment, is an intranet. Network 130 is coupled to PC 135. A floor trader 140 uses PC 135 to
10 receive and transmit information and conduct trades on the trading floor
15 145. It should also be noted that computer 120 is a server capable of running in a Win 32 platform, such as Windows 95 or Windows NT.

In general, system 100 allows customer 105 to transmit and receive information to and from trader 140. The data exchange between customer
20 105 and trader 140 includes data relating to request for quotes (RFQ), purchases of warrants or other securities and confirmation of the purchase of warrants or other securities.

Figure 2 is a Web site 200 within a browser of a brokerage house who has implemented the present invention. Section 205 contains buttons

and functions found on a standard browser. Section 210 contains links to other Web pages from this Web page that give the user increased functionality. Button 210a allows a current customer of the proprietor of this Web site to check the status of his/her portfolio. Button 210b links the 5 customer to another Web site where the customer may execute a warrants or securities trade. Button 210c links the user to another Web site where the user may learn about the brokerage house's capabilities, fees and personnel. Button 210d links users to a Web site where they can join and become customers of the proprietor of this Web site. Button 210e links the 10 user/customer to another Web site where the user/customer may contact the proprietor of the Web site via either email, telephone or regular mail service.

If the user is an existing customer and clicks on either button 210a or 210b of Figure 2, he/she will be linked to another Web site to verify that 15 he/she is authorized to transact business. Figure 3 depicts an example of a Web site to which a user is linked in order to submit the data required by computer 120 to verify that the user is an existing customer. The user enters his/her name in field 305 and password in field 310, then clicks on the submit button 315. If the user does not wish to progress with this 20 operation, he/she clicks on the cancel button 320.

Figure 4 is a Web site a customer uses to begin executing a trade. In a preferred embodiment, Web site 400 is presented to the customer when he/she clicks on link 210b on screen 200 and enters the required data of Figure 3. Web site 400 contains various fields that the customer enters to

begin a warrants or securities trade. In this example, the customer enters the name of the security or warrant he wishes to purchase in field 405. In field 410, the customer enters the number of shares in the warrant or security he/she wishes to purchase. In field 415, the customer enters the price per share he/she is willing to pay. The client application automatically multiplies the amount and price per share entered to provide the customer with a total for the purchase in field 420. The customer then clicks on button 425 to request to buy those shares, button 430 to request to sell those shares, or button 435 to cancel the transaction.

Figure 5 is a process 500 in accordance with a preferred embodiment of the present invention. At block 505, the customer logs onto the system. Typically, this involves entering a user name and password as shown in Figure 3 and computer 120 using database 125 to confirm that this is a valid customer attempting to execute a trade. At block 510, the customer enters the requisite information to execute the trade he/she desires. In some implementation of this invention, the beginning of the execution of a trade is called a request for quote (RFQ), in that the customer is requesting the trader to accept the proposed "bid" and inform the customer of the fees associated with the trade. In one example, this is done via the Web site shown in Figure 4.

At block 515, computer 120 performs a plurality of checks to determine if this customer is authorized to execute the requested transaction. For example, computer 120 pulls existing customer data from database 125 and uses the existing data to determine if the customer can

execute this trade. More specifically, computer 120 determines whether or not this customer has the credit rating sufficient to pay for the purchase, is allowed to trade in the number of shares requested or not, etc. If the customer does not have the right credentials to carry-out the requested trade
5 at block 515, computer 120 transmits a Web page informing the customer that the trade could not be executed at block 520 and the process ends at block 525.

If the customer has the right credentials to carry-out the requested trade at block 515, computer 120 transmits the trade request to PC 155 via
10 network 130 at block 530. At block 535, trader 140 has the opportunity to respond to the request from the customer. In a preferred embodiment of the present invention, trader 140 has about 7 seconds to respond to the requested trade by the customer before computer 120 revokes the request as being stale. If trader 140 does not accept the request within the time
15 allotted, the trade is voided and the customer is so informed at block 540, and the process ends at block 545.

If trader 140 accepts the trade at block 535 within the allotted time period, computer 120 sends a notice of acceptance to the customer at block 550. Also in block 550, the customer is given the opportunity to review the order and the commission charged by trader 140 to execute this trade. At
20 block 555, the customer decides if he/she wishes to accept or reject the trade. If the customer accepts the trade at block 555, then a confirmation number and a message are given to the customer at block 560, indicating that the trade has been accepted by both, the customer and trader 140, and

will be executed in due course. The process then ends at block 565. If the customer rejects the response by trader 140 at block 555, the process ends at block 570.

Figure 6 is an example of class components created as customers request service from system 100 of Figure 1. In a preferred embodiment, whenever a customer obtains access to the trading services provided by the proprietor, these three classes are created to handle the data transmitted between customer 105 and trader 140. In this preferred embodiment, if a single customer is attempting to execute multiple transactions or portfolio summaries at one time, that customer will be linked to a plurality of classes and associated objects shown in Figure 6. In an alternative embodiment, each set of classes can handle more than one customer transaction. As each customer transaction is added to computer 120, objects are also created and associated with one set of the created class components.

Computer 120 creates a plurality of classes and associated objects to handle the customers as they log into the system. Thus, sets of class components 605 and 610 are created and destroyed as the need arises. In a preferred embodiment, the individual class components in each set 605 and 610 are object linking and embedded (OLE) class types.

Class component CMS 605b supports objects that include functionality that is common to a variety of customer applications loaded onto PC 110. More specifically, this class component supports objects that allow customers to log onto system 100 and obtain access to traders 140 as well as checking the password of those customers for security purposes.

As can be seen, every customer must be verified and obtain access to traders 140 so this functionality is shared across all customers regardless of whether the customer executes a trade or not.

Class component WTS 605a contains a library of functions that

5 allow the customer to send and receive messages to and from the warrants or securities trading system, as represented by trader 140 and PC 135. Thus, the library functions within the WTS class component 605a are specific to the trading system used by trader 140. In addition, class component WTS 605a also contains library functions to register new

10 customers, obtain quotes of warrant or security prices from the trading floor 145 and maintain a transaction summary of executed buy and sell orders by recording the start and end dates of the trade, the customer reference number and system reference number.

Class component ITS 605c supports objects that act as a translator and go-between between the objects of the other class components 605a and 605b, and the trading floor 145 via ITSDL 615a. It should be noted that the class component ITS 605c is capable of operating in both a synchronous and asynchronous format.

This type of structure where the classes are created dynamically as needed is called a dynamically linked library (DLL). Instead of designing the class components to be static, the class components 605 are created dynamically to match more closely the requirements of the clients. As an example, for small time traders, the maximum amount that a smaller client may execute is limited. Additionally, depending on the pricing scheme,

small clients may be limited to fewer number of trades per day than larger clients. Larger clients may be given more options due to the volume in which they trade. Thus, classes are created with specific functions to handle smaller clients and other classes are created to process data
5 generated by larger clients.

Various preferred embodiments of the invention have been described in fulfillment of the various objects of the invention. It should be recognized that these embodiments are merely illustrative of the principles of the present invention. Numerous modifications and adaptations thereof
10 will be readily apparent to those skilled in the art without departing from the spirit and scope of the present invention.

DOCUMENT E2523260